true (1060), the KD signals a confirmation request to the user (1070) and awaits confirmation.

3) The KD authenticates the lock with its link key according to Bluetooth specification (KD challenges, LD responds) (1090, 2070, 2080, 2090, 1100). Note that the LD finds the KD's link key based on the KID, not on the KD's Bluetooth address.
4) Encryption based on the link key is taken into use (1120, 2100). The LD sends a random number to the KD (2110, 1130). The KD encodes the random number and its Bluetooth address with its secret key (1140), and sends the result to the LD (1150, 2120).
5) The LD decrypts the encrypted number and address using the public key that corresponds to the KID (2130). If the decryption is successful (meaning that the key is correct)(2140), the address matches the one the KD has (2150), and the number matches the one the LD sent (2160), and the access rights of the user allow him to get in now (2170), then the user is authorized to open the lock(2190, 1160-1180, 2200). Closing the lock again could be based on a simple timeout (lock stays open for a predefined time interval), but preferably the LD would use radio signal strength to get some measure of the distance to the KD, and when signal strength is sufficiently weak (2210), close the lock again(2220).

Preferably, public key cryptography is used to authenticate KDs to LDs, because then it is enough to store a single RSA key pair on the KD. However, it is also possible to store a separate key for each LD on the KD. This has the disadvantage that more storage is required, but the advantage that a more efficient cryptographic method (such as block cipher) can be used instead of public key cryptography. Preferably, Bluetooth combination keys should be used, since they can then easily also be used to individually authenticate LDs.

A KD contains all the tickets of its user. A ticket can be created based either on a key or another ticket. In the former case we use the term original ticket, and in the latter case we use the term derivative ticket. In literature, the term delegated certificate is also used for the same concept.

Each ticket is assigned a unique ticket identifier (hereafter TID) when creating it. A TID is useful for two things: It allows tickets to be revoked individually (by storing the TID on a blacklist on the LD) and it allows single-use tickets (by marking the ticket as single-use in its access limits and having the LD store the TID of tickets so marked on its blacklist after use). The method can be expanded to tickets with *n* uses (hereafter *n*-use or N-use tickets) by storing the total number of allowed uses in the ticket's access limits, and by adding a count to the blacklist that is increased with each access. The ticket is then refused only when the access would actually increase the count in the blacklist above the total number of allowed accesses stored in the ticket's access limits. Note that tickets with a limited number of uses should preferably also have a validity time limit (a not-after date) to allow LD's to purge obsolete information from their blacklists.

If a *n*-use ticket can be used for several LDs, which are not in contact with each other, it is possible to copy the ticket and use it several times, since the LDs cannot keep track of the total number of uses. If LDs store ticket use information, this can later be detected by combining the data in the different LDs, and the fraudulent user can then be identified. However, detection after the fact is obviously not appealing. The security can be strengthened by requiring that the KDs of the ticket receivers have an authentication token from a well-known trusted authority, that contains the KD's Bluetooth address (or a similar unique network address if another communication technology is used), digitally signed by the authority. The idea is that such tokens are only given to KDs that cannot

easily be used for copying tickets. Such KDs would of course also only allow derivative tickets to be created for KDs that have a similar authentication token.

Preferably, a CD is used for creating tickets. If the LD supports a blacklist of TIDs, the TID (that is included in the ticket) should preferably be stored on the CD, so that, if necessary, the ticket can later be revoked by adding the TID to the blacklist. If the ticket has a validity time limit, that should be stored on the CD as well, so that it also can be stored on the blacklist.

A ticket based on a key contains a LID, a TID, the KID of the granter, a link key, the public key of the receiver, access limits, the maximum number of levels of derivative tickets, limits on the derivative tickets (e.g. only one-day derivative tickets may be created) and a checksum encoded with the secret key of a user authorized to grant such tickets.

The link key of a ticket is created using a (well-known) secure one-way hash on the granter's link key and the LID.

A derivative ticket contains a LID, a TID, the public key of the receiver, access limits, the maximum number of levels of derivative tickets, limits on the derivative tickets, a checksum encoded with the secret key of the granter, and additionally the original ticket that the granter has. Note that derivative tickets may be nested to an arbitrary depth.

A KD may store and maintain a use count for a n-use ticket to keep track of how many uses the ticket has left. The use count is then increased with each use, and compared to the total number of uses in the ticket to get the number of remaining uses (alternatively, a decreasing counter could also be used). Note that when a n-use derivative ticket is used, the use count is increased for all its *n*-use nested tickets as well. This means that when a *n*-use ticket is used to create derivative tickets, the owner of the first ticket cannot know any more how many uses his ticket has at a certain time. We call tickets with this kind of shared number of uses shared tickets. Alternatively, if the derived